Hvilket sprog er hurtigst

Denne artikel forsøger at aflive forskellige myter om hvilke sprog der er hurtigst.

Den forudsætter ikke noget særligt.

Skrevet den **05. Feb 2009** af **arne_v** I kategorien **Programmering / Generelt** | ★★★☆☆

Historie:
V1.0 - 14/01/2004 - original
V1.1 - 31/01/2004 - forbedret formatering
V1.2 - 07/04/2004 - lave gruppering for at tydeliggøre resultaterne
Påstande og fakta
Et ofte stille spørgsmål blandt programmører er
"hvilket sprog er hurtigst?".

Og mange har meget bestemte meninger:
- C++ er hurtigere end Delphi
- Java er langsomt i forhold til native
- .NET er ligeså langsomt som Java
- gratis compilere er meget langsommere end kommercielle
etc.etc.

Jeg har lavet et lille eksempel til at undersøge sagen.

Der er både en integer og en floating point version.

Det er ikke nogen perfekt bencmark. Det er et meget
meget lille program.

Men:
- det er så nemt at man kan kode en version i ethvert sprog som måtte mangle hvis man vil sammenligne
- det er godt nok til faktisk at give forskelle

Lad os se nogen resultater.

Testene er lavet på en Athlon XP 2000+ (1667 MHz) med
512 MB RAM og Windows 2000 Professional.

Integer test:

```
SUN JVM 1.3.1 = 30 seconds
SUN JVM 1.4.2 = 5 seconds
BEA JRockIt JVM 8.0 = 7 seconds
IBM JVM 1.3.0 = 3 seconds
G++ 2.95 = 5 seconds
G++ 3.1 = 8 seconds
```

```
  G++ 3.2 = 7 seconds
  G77 2.95 = 3 seconds
  G77 3.1 = 7 seconds
  G77 3.2 = 7 seconds
  MS VC++ 6 standard = 6 seconds
  MS VC++ 6 /Ox /G5 = 4 seconds
  MS .NET 1.1 SDK C# = 3 seconds
  Borland C++ Builder 5.5 = 3 seconds
  Borland Delphi 6 = 3 seconds
```

Hvilket indenfor den usikkerhed der nu er må kunne opsummeres som:
gruppe 1 (3-4 sekunder) : IBM Java, MS VC++, C#, C++ Builder, Delphi
gruppe 2 (5-8 sekunder) : SUN Java, BEA Java, GCC

Floating point test:

```
  SUN JVM 1.3.1 = 23 seconds
  SUN JVM 1.4.2 = 19 seconds
  BEA JRockIt JVM 8.0 = 24 seconds
  IBM JVM 1.3.0 = 10 seconds
  G++ 2.95 = 25 seconds
  G++ 3.1 = 20 seconds
  G++ 3.2 = 20.5 seconds
  G77 2.95 = 24.5 seconds
  G77 3.1 = 20 seconds
  G77 3.2 = 19 seconds
  MS VC++ 6 standard = 20 seconds
  MS VC++ 6 /Ox /G5 = 9.5 seconds
  MS .NET 1.1 SDK C# = 18 seconds
  Borland C++ Builder 5.5 = 21 seconds
  Borland Delphi 6 = 19 seconds
```

Hvilket indenfor den usikkerhed der nu er må kunne opsummeres som:
gruppe 1 (9-10 sekunder) : IBM Java, MS VC++
gruppe 2 (18-25 sekunder) : SUN Java, BEA Java, GCC, C#, C++ Builder, Delphi

Vi konkluderer udefra kombinationen af integer og floating point resultaterne at:
* Delphi er ligeså hurtigt som C++
* de bedste JVM er ligeså hurtige som native kode
* .NET er ligeså hurtigt som native kode
* gratis compilere er sommetider ligeså hurtige som kommercielle

Husk at det kun er en enkelt simpel test. Så dem med bestemte meninger kan naturligvis fortsat have dem.

Nogle vil påpege, at der ikke indgår objekter, så at Java og C# ikke skal garbage collecte. Det er rigtigt, men eksemplet er et arithmetik eksempel og arithmetik bruger ikke garbage collection. Iøvrigt viser test normalt at garbage collection ikke forringer overall performance markant, men at den kun giver dårligere real time egenskaber.

Reelt er sprogene rimeligt ens i hastighed. Men det er der heller ikke noget overraskende i. Fordi:

* de kører på samme CPU
* compilere udviklerne studerer naturligvis hinandens arbejde og låner gode ideer til forbedring
* compiler teknologi har udviklet sig rigtigt meget de sidste 30 år med hensyn til optimering
* compilere fra samme levandør vil ofte dele compiler backend (compiler frontend = source code parsning, compiler backend = kode generering)

Den største forskel i compiler teknologi er nok mellem Java og .NET, hvor optimeringen foregår runtime, og de native, hvor optimeringen foregår på compile tidspunkt.

Og så lige en lille efternote: betydningen af sprogs hastighed er kun relevant for en meget lille del af de applikationer som skrives idag.

Det er normalt den fremmede kode (operativ system, database etc.) som bruger langt størstedelen af resourcerne idag, mens ens egen kode med high level logik betyder meget mindre. Og så betyder et sprogs hastighed ikke ret meget.

CPU kraft er billigt. Får 5000 kr. får man en superhurtig PC. Software udvikling er dyrt - rigtigt dyrt - et sted mellem 500 og 1500 kr. i timen. Hvis en applikation kun skal køre på en eller ihvertfald få PC'ere, så skal man vælge sprog efter hvad der giver kort udviklings tid og billig vedligehold fremfor efter hastighed.

Koden

Her komme kilde teksten, så kan folk selv compile og teste deres compilere på deres egen PC.

Java

```
public class Test {
    private static final int N = 1000000000;
    public static void main(String[] args) throws Exception {
        long t1 = System.currentTimeMillis();
        int sum = 0;
        for(int i = 0; i < N; i++) {
            sum = ((sum + 1) * 2 + 1) / 2;
        }
        long t2 = System.currentTimeMillis();
        if(sum != N) {
            System.out.println("Error");
        } else {
            System.out.println(N + " operations in " + ((t2 - t1) / 1000) + "
seconds");
        }
    }
}
```

C++

```
#include <iostream>

#include <time.h>

using namespace std;

const int N = 1000000000;

int main()
{
  time_t t1 = time(NULL);
  int sum = 0;
  for(int i = 0; i < N; i++) {
      sum = ((sum + 1) * 2 + 1) / 2;
  }
  time_t t2 = time(NULL);
  if(sum != N) {
      cout << "Error" << endl;
  } else {
      cout << N << " operations in " << (t2 - t1) << " seconds" << endl;
  }
}
```

Fortran

```
      PROGRAM TEST
      INTEGER N, I, SUM, T1, T2, SCALE
      PARAMETER (N = 1000000000)
      CALL SYSTEM_CLOCK(T1, SCALE)
      SUM = 0
      DO 100 I = 0,(N - 1)
        SUM = ((SUM + 1) * 2 + 1) / 2;
100   CONTINUE
      CALL SYSTEM_CLOCK(T2, SCALE)
      IF(SUM.NE.N) THEN
        WRITE(*,*) 'Error'
      ELSE
        WRITe(*,*) N, ' operations in ', (T2 - T1) / SCALE, ' seconds'
      ENDIF
      END
```

C#

```
using System;

class SpeedTest
{
    private const int N = 1000000000;
```

```
    public static int Main() {
        long t1 = DateTime.Now.Ticks;
        int sum = 0;
        for(int i = 0; i < N; i++) {
            sum = ((sum + 1) * 2 + 1) / 2;
        }
        long t2 = DateTime.Now.Ticks;
        if(sum != N) {
            Console.WriteLine("Error");
        } else {
            Console.WriteLine(N + " operations in " + ((t2 - t1) / 10000000) +
" seconds");
        }
        return 0;
    }
}
```

Delphi

```
program Test;

{$APPTYPE CONSOLE}

uses
  Windows, SysUtils;

const
  N = 1000000000;

var
  t1, t2, sum, i : integer;

begin
  t1 := GetTickCount;
  sum := 0;
  for i := 0 to (N - 1) do begin
    sum := ((sum + 1) * 2 + 1) div 2;
  end;
  t2 := GetTickCount;
  if(sum <> N) then begin
      writeln('Error');
  end else begin
      writeln(N:1, ' operations in ', ((t2 - t1) div 1000):1, ' seconds');
  end;
end.
```

Java

```
public class TestFP {
    private static final int N = 1000000000;
    public static void main(String[] args) throws Exception {
        long t1 = System.currentTimeMillis();
```

```
        double sum = 0;
        for(int i = 0; i < N; i++) {
            sum = ((sum + 1) * 2 + 1) / 2;
        }
        long t2 = System.currentTimeMillis();
        if(Math.abs(sum-1.5*N) > 1) {
            System.out.println("Error");
        } else {
            System.out.println(N + " operations in " + ((t2 - t1) / 1000) + "
seconds");
        }
    }
}
```

C++

```
#include <iostream>

#include <time.h>
#include <math.h>

using namespace std;

const int N = 1000000000;

int main()
{
  time_t t1 = time(NULL);
  double sum = 0;
  for(int i = 0; i < N; i++) {
      sum = ((sum + 1) * 2 + 1) / 2;
  }
  time_t t2 = time(NULL);
  if(fabs(sum-1.5*N) > 1) {
      cout << "Error" << endl;
  } else {
      cout << N << " operations in " << (t2 - t1) << " seconds" << endl;
  }
}
```

Fortran

```
      PROGRAM TEST
      INTEGER N, I, T1, T2, SCALE
      DOUBLE PRECISION SUM
      PARAMETER (N = 1000000000)
      CALL SYSTEM_CLOCK(T1, SCALE)
      SUM = 0
      DO 100 I = 0,(N - 1)
        SUM = ((SUM + 1) * 2 + 1) / 2;
100   CONTINUE
      CALL SYSTEM_CLOCK(T2, SCALE)
```

```
      IF(ABS(SUM-1.5*N).GT.1) THEN
        WRITE(*,*) 'Error'
      ELSE
        WRITe(*,*) N, ' operations in ', (T2 - T1) / SCALE, ' seconds'
      ENDIF
      END
```

C#

```
using System;

class SpeedTest
{
    private const int N = 1000000000;
    public static int Main() {
        long t1 = DateTime.Now.Ticks;
        double sum = 0;
        for(int i = 0; i < N; i++) {
            sum = ((sum + 1) * 2 + 1) / 2;
        }
        long t2 = DateTime.Now.Ticks;
        if(Math.Abs(sum - 1.5*N) > 1) {
            Console.WriteLine("Error");
        } else {
            Console.WriteLine(N + " operations in " + ((t2 - t1) / 10000000) +
" seconds");
        }
        return 0;
    }
}
```

Delphi

```
program Test;

{$APPTYPE CONSOLE}

uses
  Windows,
  SysUtils;

const
  N = 1000000000;

var
  t1, t2, i : integer;
  sum : double;

begin
  t1 := GetTickCount;
  sum := 0;
  for i := 0 to (N - 1) do begin
```

```
    sum := ((sum + 1) * 2 + 1) / 2;
  end;
  t2 := GetTickCount;
  if(abs(sum - 1.5*N) > 1) then begin
      writeln('Error');
  end else begin
      writeln(N:1, ' operations in ', ((t2 - t1) div 1000):1, ' seconds');
  end;
end.
```

Kommentar af borrisholt d. 16. Aug 2005 | 1

Alle de test jeg har deltaget i viser at optimeret kode er lige hurtig i c++ og i Delphi. Jeg synes du laver en fin gennemgang.

Do glemmer du en vigtig detalje : Borland er dygtigere til at skrive Compilere !!! Det ultimative bevis for dette findes i at Microsoft forud for større sastsninger på deres compiler har de altid købt en mand fra Borland.

I øvrigt har Borland uden sammenligning den laveste compile tid !!

Kommentar af simonvalter d. 15. Jan 2004 | 2

Interessant!

Kommentar af bearhugx d. 24. Jan 2004 | 3

Interessant læsning - og fik da punkteret nogle myter hos mig selv :-=)

Kommentar af skwat d. 31. Oct 2005 | 4

pussig lille sag.

Kommentar af hermandsen d. 22. Jan 2004 | 5

God artikel med en interessant konklusion! Helt sikkert værd at læse!

Kommentar af skovenborg d. 04. Feb 2004 | 6

Ja, så fik man det slået på plads. Meget interesant må man sige ;-)

Kommentar af athlon-pascal d. 15. Jan 2004 | 7

God artikel :o)
Spændende resultat.

Kommentar af xodeus d. 15. Jan 2004 | 8

Meget interessant artikel.
men eftersom man skriver mindst kilde i java og C# så er de da forholdsvis lige hurtige?

Kommentar af cronck d. 03. Feb 2004 | 9

Så er du ikke i tvivl længere

Kommentar af chrha22 d. 02. Jul 2004 | 10

Kommentar af ib-rene-cairo d. 19. Feb 2004 | 11

Overskriften "Hvilket sprog er hurtigst" er misvisende.
Testen viser jo tydeligt at det er valget af compiler, der er afgørende for hastigheden og IKKE hvilket programmeringssprog man vælger.
Konklusionen "Delphi er ligeså hurtigt som C++" er noget vrøvl. Det er kun sandt, hvis begge compilere er fra firmaet Borland. Oversigten viser klart at de fleste C++ compilere er langsommere end Borlands Delphi.

Kommentar af mikkelbm d. 15. Jan 2004 | 12

Fint lille eksempel til at illustrere forskellene.

Kommentar af trer d. 24. Jan 2004 | 13

Fin artikel, savner dog en sammenligning på strengoperationer og med varianttyper.

Kommentar af sandbox d. 15. Jan 2004 | 14

Interessant. Især konklusionen.

Kommentar af tigertool d. 24. Aug 2004 | 15

Kommentar af nlf d. 08. May 2005 | 16

Kommentar af visualdeveloper d. 15. Aug 2005 | 17

genial artikel !

Kommentar af asgerm d. 15. Apr 2005 | 18

"Hvilket sprog er hurtigst" ja men mangler der ikke nogle sprog for det her gælder jo KUN for Java, C++, Fortran, C# og Delphi, hvor er ASM, Pascal, VB og Python m.m.??